

### REMARKS

This Response is responsive to the Final Office Action mailed June 2, 2003. In that action: Claims 1-19 were pending; the rejection of Claims 1-19 from the previous Office action was maintained, those previous rejections being: Claims 1-7 and 9 were rejected under 35 U.S.C. § 102(e) as being anticipated by Eyal (U.S. Patent No. 6,389,467); Claims 8, 10-12 and 14-19 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Eyal in view of Martino (U.S. Patent No. 5,987,103); and Claim 13 was rejected under Section 103(a) as being unpatentable over Eyal in view of Martino and Suzuki (U.S. Patent No. 6,470,356).

A first Response to Final Action was filed by the undersigned on August 1, 2003. The same arguments in the below paragraphs were made. The undersigned did not, however, submit a copy of the U.S. provisional application upon which the Eyal patent claims priority. This was because the undersigned apparently incorrectly assumed that the Examiner would prefer to rely upon examining the appropriate Patent Office file for the provisional application rather than rely on our copy of the file history we obtained from the Patent Office. An Advisory Action was mailed August 26, 2003, in which our arguments were found to be non-persuasive because we did not provide a copy of the Eyal provisional application. Such a copy is now provided and reconsideration of the final rejection is hereby requested.

Claims 1-7 and 9 have been rejected as anticipated by Eyal. Eyal appears to disclose a streaming media search and continuous playback system, in which the system collects network addresses based on search criteria and plays back media resources located at some of the selected addresses in an automatic fashion.

The Final Office action relies primarily on a section in the Eyal reference located at column 2, line 43 through column 3, line 9 for disclosure of the limitations of “compiling a download schedule” and “a file download device, which based on the download schedule,

automatically accesses . . . .” A rejection of these claims based on Eyal is made under 35 U.S.C. § 102(e). The Eyal patent issued on May 14, 2002, having been filed on May 2, 2000, and relying for priority on U.S. Provisional Patent Application No. 60/177,786 (a copy of which is now provided), filed on January 24, 2000. The present application was filed on March 2, 2000 (in between the 2 Eyal filings) so the teachings located in the Eyal patent can only possibly qualify as an anticipation if they are located in the provisional patent application. In other words, material that first appeared in the non-provisional application filed on May 2, 2000 cannot be utilized in a §102(e) rejection of the present invention.

The specification of U.S. Provisional Patent Application No. 60/177,786 has been compared to the specification of the Eyal patent (U.S. Patent No. 6,389,467). It appears that the entire Summary of the Invention section running from column 1, line 51 through column 8, line 43 does not appear in the provisional application. In addition, the following paragraphs in the Eyal patent could not be found in the provisional application: the last two paragraphs beginning in column 10, the first three paragraphs and paragraphs 5 and 6 in column 11, all of the new paragraphs in columns 18 and 19, all but the last paragraph beginning in column 20, all but the fifth and sixth new paragraphs in column 21, the fifth paragraph in column 22, all but the first, third, and fifth paragraphs in column 32, all of columns 33-37, and the first new paragraph in column 38. Of course, none of the claims in the Eyal patent are found in the provisional patent application either.

As can be appreciated, the primary portion of the Eyal reference relied upon by the Examiner to reject Claims 1-7 and 9 is not available as an anticipatory teaching since it is not in the provisional application. Numbered paragraph 4 on pages 2 and 3 of the Final Office Action does state that “this limitation and others that are listed in the claims that are rejected by Eyal are further pointed out in the following sections.” Of the following sections that are listed, only one

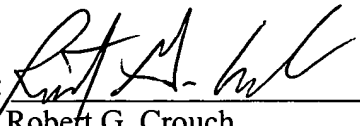
appears in the provisional application. This is the section described in the Final Office Action as column 15, line 17 through column 16, line 54. This section does appear to be in the provisional application. It is respectfully believed, however, that this section does not anticipate the rejected claims. There is no selection interface which is configured to receive and process selections for compiling a download schedule, nor is there a file download device, which based on the download schedule, automatically accesses the remote site and downloads the selected multimedia file. There is also no disclosure in the provisional application of the limitation the Examiner points to in columns 2 and 3 of the patent of the network interface signaling a request to a network service module and receiving addresses that match the search request. For all these reasons, it is respectfully submitted that Claims 1-7 and 9, and all claims dependent thereon (including Claim 8) are patentable.

Claim 10 has been rejected as obvious in light of a combination of Eyal and Martino. Since Claim 10 has somewhat similar limitations (“compiling a download schedule based on the received inputs, wherein the scheduling includes a description of the multimedia files selected, day and time for download, and download information” and “based on the inputs received through the interface, accessing and downloading over the data network, the selected multimedia files from selected remote sites”), and since neither Martino nor the provisional application upon which the Eyal patent is based contain such teaching, it is respectfully submitted that Claim 10 and all claims dependent thereon (including Claims 11-19) are patentable.

Based upon the foregoing, Applicants believe that all pending claims are in condition for allowance and such disposition is respectfully requested. In the event that a telephone conversation would further prosecution and/or expedite allowance, the Examiner is invited to contact the undersigned.

Respectfully submitted,

MARSH FISCHMANN & BREYFOGLE LLP

By:   
\_\_\_\_\_  
Robert G. Crouch

Registration No. 34,806  
3151 South Vaughn Way, Suite 411  
Aurora, Colorado 80014  
(720) 562-5506

Date: September 4, 2003



RECEIVED

SEP 11 2003

Technology Center 2100

PATENT APPLICATION SERIAL NO. \_\_\_\_\_

U.S. DEPARTMENT OF COMMERCE  
PATENT AND TRADEMARK OFFICE  
FEE RECORD SHEET

02/01/2000 HHAGSHHT 00000104 232415 60177786

01 FC:114 150.00 CH



RECEIVED

SEP 11 2003

Technology Center 2100

file:///c:/APPS/preexam/correspondence/1.htm



Bib Data Sheet



UNITED STATES DEPARTMENT OF COMMERCE  
Patent and Trademark Office

Address: COMMISSIONER OF PATENTS AND TRADEMARKS  
Washington, D.C. 20231

<b>SERIAL NUMBER</b> 60/177,786	<b>FILING DATE</b> 01/24/2000 <b>RULE</b> -	<b>CLASS</b> -	<b>GROUP ART UNIT</b> -	<b>ATTORNEY DOCKET NO.</b> 24679-701
<b>APPLICANTS</b> Eviv Eyal, San Francisco, CA ; George Aposporos, San Francisco, CA ;  ** CONTINUING DATA *****  ** FOREIGN APPLICATIONS *****				
<b>IF REQUIRED, FOREIGN FILING LICENCE GRANTED ..</b> ** 02/28/2000				
Foreign Priority claimed <input type="checkbox"/> yes <input type="checkbox"/> no 35 USC 119 (a-d) conditions <input type="checkbox"/> yes <input type="checkbox"/> no <input type="checkbox"/> Met after Allowance		<b>STATE OR COUNTRY</b> CA	<b>SHEETS DRAWING</b> 16	<b>TOTAL CLAIMS</b> - <b>INDEPENDENT CLAIMS</b> -
Verified and Acknowledged Examiner's Signature _____ Initials _____				
<b>ADDRESS</b>  Wilson Sonsini Goodrich & Rosati 650 Page Mill Road Palo Alto, CA 94304-1050				
<b>TITLE</b>  STREAMING MEDIA SEARCH AND PLAYBACK SYSTEM				
<b>FILING FEE RECEIVED</b> 150	FEES: Authority has been given in Paper No. _____ to charge/credit DEPOSIT ACCOUNT No. _____ for following:		<input type="checkbox"/> All Fees <input type="checkbox"/> 1.16 Fees ( Filing ) <input type="checkbox"/> 1.17 Fees ( Processing Ext. of time ) <input type="checkbox"/> 1.18 Fees ( Issue ) <input type="checkbox"/> Other _____ <input type="checkbox"/> Credit	

**RECEIVED**

SEP 11 2003

Technology Center 2100

01-27-00

A/P/200

JC662 U.S. PTO  
01/24/00Approved for use through 04/11/98. OMB0651-0037  
Patent and Trademark Office, U.S. DEPARTMENT OF COMMERCE**PROVISIONAL APPLICATION COVER SHEET**

This is a request for filing a PROVISIONAL APPLICATION under 37 CFR § 1.53(e)

Express Mail label number EL322095737US Date of Deposit January 24, 2000  
I hereby certify that this paper or fee is being deposited with the United States Postal Service  
"Express Mail Post Office to Addressee" service under 37 CFR § 1.10  
on the date indicated above and is addressed to the Assistant Commissioner for Patents, Washington, DC 20231.Donna L. Hengst

Name of person signing

Donna R. Hengst  
SignatureJC553 U.S. PTO  
60/17786  
01/24/00

50177786-012400

Docket Number		24679-701	Type a plus sign (+) inside this box →	+
<b>INVENTOR(s)/APPLICANT(s)</b>				
LAST NAME	FIRST NAME	MIDDLE INITIAL	RESIDENCE (CITY AND EITHER STATE OR FOREIGN COUNTRY)	
EYAL	Eviv		San Francisco, CA	
APOSPOROS	George		San Francisco, CA	
<b>TITLE OF THE INVENTION (280 characters max)</b>				
STREAMING MEDIA SEARCH AND PLAYBACK SYSTEM				
<b>CORRESPONDENCE ADDRESS</b>				
WILSON SONSINI GOODRICH & ROSATI 650 Page Mill Road Palo Alto, California 94304-1050 Telephone: (650) 493-9300 Facsimile: (650) 493-6811				
<b>ENCLOSED APPLICATION PARTS (check all that apply)</b>				
<input checked="" type="checkbox"/> Specification	<i>Number of Pages</i> <u>39</u>		<input type="checkbox"/> Small Entity Statement	
<input checked="" type="checkbox"/> Drawing(s)	<i>Number of Sheets</i> <u>16</u>		<input type="checkbox"/> Other (specify) _____	
<b>METHOD OF PAYMENT (check one)</b>				
<input type="checkbox"/> A check or money order is enclosed to cover the Provisional filing fees.		PROVISIONAL FILING FEE AMOUNT (\$)		\$150.00
<input checked="" type="checkbox"/> The Commissioner is hereby authorized to charge filing fees and credit Deposit Account Number: <u>23-2415</u>				

The invention was made by an agency of the United States Government or under a contract with an agency of the United States Government.

☒

No.

☐

Yes, the same of the U.S. Government agency and the Government contract numbers are: \_\_\_\_\_

Respectfully submitted,

SIGNATURE

Van MahamediDate: January 24, 2000TYPED or PRINTED NAME Van MahamediREGISTRATION NO. 42,828

(if appropriate)

☐

Additional inventors are being named on separately numbered sheets attached hereto.

**PROVISIONAL APPLICATION FILING ONLY**



**RECEIVED**

SEP 11 2003

Technology Center 2100

60177786-012400

UNITED STATES PATENT APPLICATION

OF

AVIV EYAL AND GEORGE APOSPOROS

FOR

**STREAMING MEDIA SEARCH AND PLAYBACK SYSTEM**

PREPARED BY WILSON SONSINI GOODRICH & ROSATI



## BACKGROUND OF THE INVENTION

### Field of the Invention

This invention relates to the field of streaming media content search and playback over a network. In particular, the invention relates to a computer system that enables a  
5 continuous streaming media playback from a distribution of sites available over a network such as the Internet.

### Description of the Related Art

Computers currently can access streaming media on the Internet. Streaming media available on the Internet include, for example, music, video clips such as movie trailers,  
10 home movies, and animation.

Users locate streaming media on the Internet by manually selecting links. Typically, users browse the media sites that contain numerous sub-links. Users sometimes select through a chain of links to locate a desired media on a media link. Once located, the desired media link may or may not contain the desired media.

15 Some services provide media search engine capabilities. Users may enter a search request for selected media creations by an artist. The media search engine then displays links to categories and/or sub-links of media that are determined to match one or more criteria in the search request set forth by the user. The determination of which links should be displayed in response to the search request is dependent on the algorithm used in  
20 by the search engine. Typically, links displayed to users of current search engines are not subject to a determination of the quality or availability of the media associated with the media links. Further, the search results are outputted to the user as a display of links for the user's selection.

Many Internet streaming media outlets provide a limited number of source nodes. The sites can be unreliable when the number of users accessing the site become congested.

## DETAILED DESCRIPTION

### A. System Overview

5 According to an embodiment, a system is provided comprising a media search engine. The media search engine may be used to create a database of links to media files. The links may be structured according to predefined categories and/or user-defined search criteria. A client terminal includes a media player to automatically access one or more media files using the corresponding links. The media player then plays back media  
10 contained on the media files.

Among other advantages of the invention, the user terminal accesses media files at various sites on a network, without requiring users to manually select media links. For example, user-terminals may output music to a user by automatically accessing one or more Internet sites containing media files. The music is outputted without requiring users' to view and select links to sites containing the media.  
15

In contrast to embodiments of the invention, using other systems to search for Internet files containing media can be a distracting and time-consuming experience for an end user. In many instances, such a search will yield a series of links on a directory or web search page. A user may have to click on each individual link, one at a time, to play  
20 each individual media file. The selected media file may be broken and unavailable to deliver media content. Even if the number of broken links is not high, the user must still click on the links one at a time to activate each media file, providing at best a stop-and-go experience.

50477786-0124000

In one embodiment of the invention, a user terminal is able to receive continuous media streaming from multiple sites on the Internet. Multiple sites may be accessible to enable the user terminal to receive streaming media without any interaction required from an end user other than signaling a request to receive streaming media. The user terminal  
5 automatically accesses media links containing media using a media playback application.

The media playback application may be controlled by one or more server-side modules. In one embodiment, the media playback application on the user terminal interacts with one or more play-lists generated by server side modules. The play-lists contain media links for the media playback application. The media links may be structured  
10 or ordered in the play-lists. The play-lists may be generated automatically by back-end modules and/or manually by editors. The play-lists may also be generated by end users.

The media playback application may also interact with one or more server side search modules to access media links on the network. The media links may be automatically selected based on, for example, a search criteria from the end user.

15 Embodiments of the invention provide a system to search and playback media accessible over a network. In one embodiment, a media search engine is provided to enable users to request media output based on a criteria set forth in a search request. The media search engine is able to efficiently locate streaming media on the network that matches criteria set forth in a search request. The system provides continuous playback of  
20 media found on multiple sites of the network. For example, a user may search for media creations by a specified artist. The system locates one or more sites on the Internet containing media creations from the specified artist. The system enables the user terminal to automatically and continuously play back media creations available on the Internet sites.

Further, a backend system under an embodiment of the invention minimizes possibilities of broken links and mismatched search results. The backend system may also be used to perform manual and/or programmatic quality check of the media associated with each link.

5 Further, a search engine under an embodiment of the invention employs an existing Internet web browser software component on the back-end to perform searches and indexing of web resources. Server-side modules may combine to control the browser in locating media links and media sites containing media content. As a result, the media search engine under this embodiment is efficiently implemented, using existing resources  
10 on the back-end system.

Among other advantages, embodiments of the invention enable streaming media from multiple media links to be automatically played to users. Embodiments of the invention also employ a scalable and distributed architecture. Scalability in this sense means that the service is available to a large (thousands or more) audience of simultaneous  
15 listeners or viewers while minimizing bottlenecks caused by congestion. Another advantage of a distributed architecture is that the unavailability of one media site, or of one or more media on the media site, does not preclude the user terminal from receiving media from another site. As a result, users are ensured a continuous listening or viewing experience.

20 Further, streaming media may be continuously outputted to users from multiple sites on the Internet based on personalized criteria set forth by users. The criteria may be set forth in one or more requests by an end user. The end user may experience media continuously outputted from multiple sites, based on only one request from the end user. This allows a user to request media through actions such as clicking requests through a  
25 user-interface.

An embodiment of the invention enables users to share streaming media experiences with other end users. For example, users may share play-lists containing links to multiple Internet sites. This enables individuals to create media programs of streaming media using multiple sites on the Internet. For example, play-lists may be shared among  
5 end users using a host web site, or emails.

In an implementation of an embodiment, a user terminal may transmit a search request from an end user to one or more modules on a server. A client side playback module, one or more server-side modules, or a combination of client and server side modules combine to access the user terminal to a site on the Internet that contains media  
10 content immediately available for loading and playback. The response to the search request is media output through the user terminal. The media content is outputted from the user terminal without any additional action on the part of the end user after the initial search request. Once media from one site is completed, the playback module automatically enables the user terminal to access and playback media located on another Internet site.  
15 As a result, an embodiment enables the user terminal to output continuous streaming media to an end user, where the media outputted is accessed from multiple Internet sites.

As used herein, a module includes a program, a subroutine, a portion of a program; a software component or a hardware component capable of performing a stated task or function. A module can exist on a hardware component such as a server independently of  
20 other modules, or a module can exist with other modules on the same server or client terminal, or within the same program. As used herein, media refers to audio, video, images, animation, text or a combination of the former. Video includes still images, graphic presentations, as well as combination of images, such as home movie clips.

Embodiments of the invention may be implemented on the Internet. Other  
25 embodiments may be implemented on any network that carries digital information, such as

local-area networks (LANs), Wide Area Networks (WAN), Extranets, Intranets, Internet, and wireless networks, or networks utilizing wireless transmissions. An example of a network for use with an embodiment of the invention includes a network operating under a transmission control protocol/Internet protocol (TCP/IP). Embodiments of the invention may also be employed on proprietary WANS, such as America Online™. Thus, discussion of embodiments employed on the Internet are exemplary, and equally applicable to other types of networks described above.

FIG. 1 illustrates a process for use with a system to search for and playback Internet streaming media, under an embodiment of the invention. In one application, the process is performed on architectures described and illustrated with FIGS. 2 and 3. While the process is described with reference to an integral system, one or more modules or components in FIG. 3 may operate independently of other steps, components or modules and can be implemented in different systems and architectures. Further, steps mentioned with FIG. 1 may be performed concurrently with one other, or in an order different than shown in FIG. 1.

In step 110, a system builds a database of addresses (Universal Resource Locations) for Internet media sites. Media sites include web pages on the Internet that allow Web users to access streaming media. The media sites may be located through a media search engine, as described elsewhere in this application. An exemplary process for identifying media sites under an embodiment of the invention is provided with FIG. 4.

Each media site may access media through one or more media links available at the site or through other means. The media links identify Web resources having media content. These Web resources may include a file of arbitrary type. Examples of file types include Multipurpose Internet Mail Extension (MIME) types such as MOV, JPG, or RAM. The file is available for loading, browsing or playback on the World Wide Web. Each

6017786-012407

media link may be either an internal or external link relatively to that particular media sites. An internal media link on a Web-site may correspond to a URL that identifies a Web resource located on the Web domain, host, property or server of that site. An external media link on a media site identifies a Web resource that is not located on Web domain, host, property or server of that media site.

In step 120, the system identifies and stores in a database media links (URLS) for each media site. An exemplary process for identifying and storing media links on individual media site stored in a database of media site is provided with FIG. 5.

In step 130, each media link is verified. The media link is verified to contain media that is available for playback for users. Thus, broken links, inoperational or unavailable media are precluded from being verified.

In step 140, metadata information is extracted from each media link. Preferably, metadata information is extracted from each verified media link. In an embodiment, metadata may also be added to a list or database of extracted metadata. Additional metadata may be added using, for example, manual interactive editing and an editor interface (see for example, editor interface module 275 in FIG. 2). Examples of metadata information include (with an exemplary data structure type associated with each media link in parenthetical): identification (Integer), author (String), duration (String), media URL (URL), source web site (URL), media type (Integer), rating (Real number), number of votes (Integer), verification status (Boolean), edited status (Boolean), genre type (Index into a genre database table), play-list genre status (Boolean), mix (index into mixes database table), play-list mix status (Boolean), mood (index into moods database table), description (String), and play-list mood status (Boolean). One or more of these types of metadata may be extracted from the media links or from the actual media file. For example, a media link to a web resource may be extracted for identification, duration,

author, and source Web site. Similarly, one or more of these types of metadata may be added to the extracted metadata information. For example, genre type and description information may be added to the extracted metadata information.

In step 150, the system creates media play-lists using media database for  
5 predefined categories. In an embodiment, verified media links are structured into play-lists, such as described with FIG. 8.

In some embodiments, links to streaming media commercials may be inserted into the play-lists in various locations between media clips. These commercials are targeted to the audience likely to listen to the media available on the play-list. The commercial may  
10 be produced and broadcast from distributed sources, or from web server module.

In step 160, a playback interface is provided. The playback interface causes the media player component on the user terminal to play media associated with media links in each play-list. The playback interface may include features to manipulate play-lists, or to switch between play-lists. For example, the playback interface may allow for a user to  
15 skip media or web resources until a preferred media or web resource is located. The playback interface is a software or hardware application that is executed on the user terminal. The playback interface may be packaged as a web application, dynamically accessible through a Web server module, or be packaged as a desktop application software.

In an embodiment, a playback interface module includes a streaming media clips rating system that allows users to rate each clip as it is played back. The back-end module rating system uses these votes to generate rated play-lists that are available through the playback Interface for playback.

Further, the playback interface module may include a system to allow users to send  
25 Internet e-mail notifications to one or more email addressees regarding a media clip, or to



00177286-012400  
001210-98222109

send continuous streaming media programs containing multiple media clips from multiple network sources. Recipients may initiate the playback module by selecting one or more links contained in the email. Selecting a link from the email initiates the play back module on that recipient's terminal, causing the play back module to play back the media clip or the programming referred to by the sender.

The playback interface includes user interface elements that allow users to define and execute search criteria for media playback.

FIG. 2 is a block diagram illustrating an architecture of a system 200, under an embodiment of the invention. The system is shown to link a user terminal 210 with media that is accessible on the Internet 220, including the World Wide Web 215. Other embodiments of the invention may operate with different types of networks.

The user terminal 210 includes any network enabled multimedia computing platform. In particular, user terminal 210 includes any Internet enabled multimedia computing platform. Examples of computing systems for user terminal 210 include personal computers (PC), personal digital assistants (PDA), smart phones, and Internet enabled televisions and radios. The multimedia capability is manifested in the availability of a steaming multimedia playback software and or hardware component. Internet enabling means that the platform can access information over the Internet. In an embodiment, user terminal 210 runs the media location and playback interface module 270 that is accessible over the Internet. A communication channel 212, such as a phone line, wireless medium, or DSL line, is used to couple the user terminal 210 to the Internet. Alternatively, the playback module may be preinstalled on the client terminal. Under both configurations the playback module access media play-lists that are stored on an Internet Web server.

A back-end database management system 245 is provided to maintain information used in providing media searching and playback to user terminal 210. The database management system 245 receives information from modules, including server-side modules that communicate with user terminal 210. In an embodiment, modules used to provide media search and playback capabilities to user terminal 210 include a media search module 230, an automatic verification and extraction module 255, an editor module 250, a play-list generator module 260, and a web server module 270. The modules may communicate with an interface of user terminal 210.

Under an embodiment, this communication is implemented using media play-lists on the web server module 280.

The modules may also communicate with software applications or components on the user terminal, such as a Web browser application or a Streaming Media player component in a manner that will be described below.

In an embodiment, media search module 230 includes a media directories meta-crawler module 234 and a media search engine 238. The meta-crawler module 234 and the media search engine 238 may be operated independently and concurrently of one another. The meta-crawler module 234 conducts a general search of the Internet 220 to locate media sites. Media sites may include web pages that are likely to contain web resources, media links to web resources, or links to other web pages that have such media links and/or Web resources. The meta-crawler module 234 adds the address or location of each found media site to a media site table 243 maintained by database management system 245. The media site table 243 may list media sites that identify a URL for each web page located by meta-crawler 234.

In one embodiment, the entire media site table 243 is programmatically generated by meta-crawler module 234, without any manual or interactive human input. In other

embodiments, an editor module 232 may interface with database management system 245 to manually input a URL for one or more of the media sites into the media site table 243. Another embodiment may substitute editor module 232 for meta-crawler 234, so that the media subdirectory manually receives a URL for each media site.

5 The media search engine 238 accesses the media site table 243 maintained by database management system 245. The media search engine 238 identifies media links to Web resources on each media site provided in the media site table 243. In an embodiment, media search engine 238 contacts each site in the media site table 243 to locate media links. The media search engine 238 then stores the addresses of each media link in the database management system 245. In an embodiment, a URL of each media link is stored in a portion of a media and metadata table 247.

10 An automatic media verification and metadata extraction (AMVME) module 240 accesses the portion of media and metadata table 247 that contains URLs to the media or media links. The AMVME module 240 verifies each media link in media and metadata table 247. The media links are verified to contain web resources matching a criteria defining media. For example, each media link may be verified to contain a combination of audio or video, rather than be only a text document. In addition, the media links are verified as available for playback by users, to avoid broken or old links being maintained by database management system 245.

15 20 The AMVME module 240 also extracts metadata from the Web resource associated with each media link in the media and metadata table 247. Preferably, AMVME module 240 extracts metadata from verified media links. The AMVME module 240 may visit each media link on the Internet to extract metadata information, as well as verification information. The metadata extracted pertains to information available from 25 the Web resource or about the Web resource on the media link. Examples of metadata that

may be extracted by media extraction module 255 include information such as the author, duration, name, description text, broadcasting and playback quality of the media content and frame size and display resolution for images, video and home movie clips. For example, a media link may be associated with a web resource that is an audio media.

5 Metadata that may be extracted from the media creation may include the artist name, the name of the media creation, length and audio/video quality. In an embodiment, media extraction module 255 also verifies that the media is available for playback from the media site. The AMVME module 240 may access database management system 245 to store verification and metadata information in media and metadata table 247.

10 In an embodiment, a metadata editor interface 275 is included in the system 200. The metadata interface 275 accepts manual entry from an editor pertaining to metadata of the Web resource associated with each media link. The metadata interface module 275 may access one or more media links in the media and metadata table 247 to allow manual inspection of each web resource for metadata information. An editor operating metadata  
15 interface module 275 transmits a media streaming request to have the media of the Web resource replayed for inspection on a terminal. The metadata editor interface 275 then allows for additional metadata to be stored in media and metadata table 247. Preferably, the additional metadata information includes metadata that is not programmatically available from the media link containing the web resource. For example, metadata editor  
20 interface 275 may be used to add information to media and metadata table 247 information such as genre of the Web resource, description of the Web resource, and system predefined information, such as mood and mix, that are found applicable by the editor to the Web Resource.

A play-list generator module 260 generates a plurality of play-lists based on  
25 information in the database management system 245. In an embodiment, play-list

00127785-0124400

generator 260 accesses media and metadata list 247 for URLs to media contained on stored media links. The play-list generator module 260 may create play-lists 284 from predefined categories characterized by information stored in the database system for media links and metadata stored in table 247. Play-lists 284 are stored on Web server module  
5 280.

Under one embodiment, the web server module 280 includes a media location and playback application. The user terminal 210 interfaces with the media location and playback application through the Internet. The user launches the playback application by clicking a link on one or more web Sites. Under another embodiment, the playback  
10 application is pre-installed on the user terminal.

The playback application accesses the web server module 280 to load media play-lists that are stored on it. The playback application reads Media URLs and Metadata stored in one or more play-lists. This information is used to playback continuous media from the play-lists to the user.

15 The media location and playback application may output or playback media processed by the back-end system and stored in the media and metadata table 247 upon receiving a request from user terminal 210. For example, music may be outputted to the user terminal 210 continuously in a manner that resembles a jukebox, Disk Jockey Mix or a radio station.

20 An interface of the user terminal 210 enables users to skip playback of media clips, or to switch categories. For example, a user on user terminal 210 may select to hear Jazz programming, and then switch to a genre of classical music. The user may also control playback settings such as volume, pause, seek and retrieve additional media clip information, skip songs, or replay certain songs being automatically played. The user may  
25 also control and/or customize the creation of play-lists using the interface.

CONFIDENTIAL

In an embodiment, the playback application programmatically controls a streaming media multimedia software or hardware component to perform the actual streaming of the media digital bits to the user terminal's multimedia output device (such as video display and speakers hardware). The application contains functionality that responds to software events generated by the streaming media component. For example, a playback error generated by the streaming component may result in the application instructing the component to play another media file. In another example, the application determines and initiates playback of a media clip in response to the component reporting that the currently playing media has finished. The application may contain user interface elements that allow users to issue media playback commands. These commands are dispatched by the application to the component that implements the playback command for the currently played media.

In an embodiment, the playback application works in combination with functional commands on a web site. The functional commands of the web site may be made available to the user through a user-interface on the user-terminal.

In an embodiment, a categorization module 290 accesses media and metadata table 247 to add metadata and to categorize music media associated with media links in media and metadata table 247. The automatic process generates metadata such as music genre by consulting with information stored in other records in media and metadata table 247. For example, the module can automatically set the genre metadata information for all media creations available in the table, for a given artists, according to genre metadata entered for one or more media creations by the same artists. This process greatly contributes the efficiency and scalability of the back-end system.

FIG. 3 is high-level system software components diagram for the system 300, under an embodiment of the invention. The diagram shows how software components may

be written, deployed and interact to provide the functionality described by system 2000.

The components of system 200 may be described as a three-tier architecture. Components are written to spec and deployed to a backend tier, a middle tier, and a front tier. The backend tier includes the database management system 245. The database management system 245 includes a database 345 and a backend interface module 355. The backend interface module 355 may be provided with, for example, a Microsoft SQL Server system (MS SQL).

The middle tier includes modules that communicate with backend interface module 355. The middle tier may include a media sites manager 360 and a media manager 365 software components. The media sites manager 360 and the media manager 365 each independently communicate with backend interface module 355. The media sites manager 360 components exposes a programmatic interface 362 to communicate with modules and components in the front tier. The media manager 365 includes a first media manager interface 366 and a second media manager interface 368.

The front tier includes a media site module 330 and a media module 340. The media site module 330 communicates with site interface 362. The media site module 340 communicates with the first and second media manager interfaces 366 and 368. The first and second media manager interfaces 366 and 368 communicate with the media module 340. The media site module 330 includes a front-end interface 332 to a directory meta-crawler 310 and a media search engine 312 modules. The media site module 340 includes a front-end interface 342 to the media search engine 312, an editor interface module 314, and an automatic verification module 316. The directory meta-crawler 310 crawls Internet media directories web sites. The links to media web sites are handed over to the MediaSite module 330 for storage in the database. The media search engine 312 searches for media

links on Web sites provided by the MediaSite module 330, these links are transferred through Interface 342 on the Media module 340 for storage in the database module 345.

The editor interface module 314 obtains media link for editing from the Media module 340, using Interface 342 and loads the media for editorial playback from the Internet. The editors provide metadata for media that are added to the database by the Media module 340.

The verification module 316 examines media files or Web resources accessed through each media link and updates metadata regarding media availability in the database using Media module 340. This module also extracts metadata from Internet media and updates this in the database using Media module 340. The module queries the database for a batch of media records using the Media module 340 and automatically verifies and extracts metadata for the Internet media represented by these records.

With respect to communications from the backend tier to the front-end tier, database management system 345 of the backend tier provides records to the system 300. Each record or record set is disconnected from tables or databases of record(s). Disconnected records are transmitted from the backend tier to the front-end tier as active database objects (ADO) Disconnected record sets.

With respect to communications from the front-end tier to the backend tier, each disconnected record can be updated in the database by any components on any tier. Updated records are transmitted to the database management system 345 in the form of record set update operations. In an embodiment, directory meta-crawler 310 sends URLs to be added to records in database 345 to media site module 330 using an asynchronous method calls. The media search engine 312 transmits to media site module 330 using a get search method call for batch sites of URLs. The media search engine 312 uses an asynchronous method call to add media links and metadata associated with media links.



10 20170901 012100

The components and all tiers expose programmatic interfaces that contain callable methods using the MS DCOM (distributed component object model) software component technology. Communication between the tiers is also implemented using method calls on these components. The components are deployed in front, middle and back tier hardware systems. Alternatively, The components may be developed and deployed using the MS COM+ components technology. Using this technology, a COM+ In Memory Database system (IMDB) proxies and caches tables of the back-end database module 245. This process speeds up the search and editorial process. COM+ services such Queued Components may used to implement asynchronous method calls exposed on Interfaces 362 and 366.

#### B. Media Search Engine

FIG. 4 illustrates a process for a component of an Internet media search module, under an embodiment of the invention. A process such as described with FIG. 4 may be used to build a database of media sites, where each media site includes media links and/or links to other media sites. In an embodiment, the process of FIG. 4 is applicable to meta-crawler module 234 in system 200 (FIG. 2).

The process of FIG. 4 provides for extracting URLs of media sites from a web pages directory. Examples of a web directory for use with an embodiment of the invention includes directories on Web sites such as Yahoo.com® and Lycos.com®. The flow process of FIG. 4 is a backend operation that is unobservable to a user of user terminal 210. Preferably, the flow process of FIG. 4 is an automatic or programmatically controlled process that does not require human interaction.

In step 410, a directory home page is added to a searched-pages data structure. The searched-pages data structure maintains. A similarly structured parsed-pages data structure

00127786-012100

is also maintained to hold URL of pages already processed by the module. The parsed-  
pages data structure indicates whether a home page web directory was previously parsed  
by the process. The searched-pages and parsed- pages data structures are keyed or indexed  
by URL and they support querying for existence of a given URL in them. Examples of  
5 keyed or indexed data structures include database tables and hashtables.

In step 410, the parsed-pages data structure is empty, indicating that the directory  
home pages in the searched-page data structure have not been parsed.

In step 420, a determination is made as to whether the searched-pages data  
structure is empty. If the determination is affirmative, the flow process is done. This  
10 occurs when the process has parsed all the Internal web pages in the directory. If the  
determination is negative, a current page link is called from the searched-pages data  
structure in step 430. The current page is then loaded into memory and parsed. Parsing  
means loading and reading the HTML(or equivalent) code of the web page so its content is  
accessible and in a machine-readable format.

15 In an embodiment, the page is parsed using an HTML parser component. An  
example of a HTML parser is a Web Browser. Thus, the current page may be parsed  
using a Web Browser component. Specifically, step 440 of the process may be  
implemented using an application program interface (API) that is exposed by the Web  
Browser component. In this context, the web Browser component is configured and used  
20 in a back-end server process with no visible presentation area or end user.

In step 440, all links to media sites on the currently parsed-page are determined  
using the parser. In step 450, all new external page links are added to the media sites  
database. An example of a web-page data structure is provided with media site database  
table 243 (FIG. 2). The database stores the URL of the media sites and not the sites

themselves. New external page links implies media sites that are not already indexed or present in the media site database.

In step 460, all URLs also link to internal links found on the currently parsed page are added to the web pages data structure, provided that the URL in question is (i) not already existing in the searched pages data structure and (ii) not already existing in the parsed pages data structure. In step 470, the currently parsed page is moved to the parsed pages data structure, and the flow process returns to step 420. This process adds the URL of all the media sites indexed by the directory to the media sites database.

FIG. 5 illustrates another component of an Internet media search module, under an embodiment of the invention. A process such as described with FIG. 5 may be used to identify and store media links to Web resources that are accessible from one or more Web site. The process of FIG. 5 may be used in conjunction with a process such as described with FIG. 4. In an embodiment, the process of FIG. 5 is applicable to media search engine 238 in system 200 (FIG. 2).

In an embodiment, the flow process of FIG. 5 is a backend operation that is unobservable to a user of the user terminal. Preferably, the flow process of FIG. 5 is an automatic or programmatically controlled process, conducted periodically. For example, media links may be identified and stored under a flow process such as shown by FIG. 5 every few days or weeks. The duration between executions of the flow process of FIG. 5 maybe referred to as an idle period.

The flow process of FIG. 5 assumes access to a media sites database store. The database includes URLs to each media site. An exemplary database of media sites includes media sites and metadata table 243, described with FIG. 2. Each record contains a URL field for media site and a field indicating the last date, if any that the process described in FIG. 5 lastly processed the web site at the URL in the URL field. Reference

to a media site that is parsed implies that the media site was programmatically examined for media links to web resources, and for links to other media sites using a process such as the one described in FIG 5.

In step 510, MIME types are determined for Web resources. Examples of MIME  
5 types that can be selected for step 510 include JPEG, MOV, .RAM and WAV.

In step 515, a record for a media site is fetched or received from the database. A condition of the media site received is that the media site was not parsed by the process described here during the idle period. This condition may be specified by checking, for example, the date field associated with the record. For example, a date field may indicate  
10 when the media site was previously parsed.

A determination is made in step 520 as to whether a record and a URL was received in step 515. If no URL was received, the system interprets that all media sites in the database have already been parsed during the idle period. If a URL is received, the system in step 525 adds the URL of the media site to a URLS data structure of media sites  
15 to be processed. The URLS data structure of unparsed media sites may be indexed or keyed. For example, the URL data structure may be a list, or a hashtable software data-structure.

In step 526, the last search field of the record fetched in step 515 is updated with the current date to indicate that the media site is parsed. In an embodiment, the field  
20 corresponds to a date in which the last parsing occurred.

In step 530, a URL in the data structure of unparsed media sites is fetched or received. If in step 535, a determination is made that the URL data structure is empty, the system returns to step 515. As will be further described, the flow process returns to step 515 only when step 570 is completed. If a determination is made that the URL data  
25 structure is not empty. Thus, steps 510-545 allow the flow process to distinguish between

when a media site is being parsed for the first time, or has been previously parsed by the process during the idle period.

In step 545, media links on the media site fetched in step 530 are extracted from the HTML code of the page fetched in step 530. The media links are associated with web resources on that media site. In step 545, the media resources may be in any MIME format recognizable as media. In an embodiment, the currently parsed page is parsed using an HTML parser. An example of an HTML parser is a web browser. Thus, the media links may be extracted using a web browser. Specifically, the media links may be extracted using an application program interface (API) provided by a web browser software or hardware component. The web browser may be configured to perform this task efficiently by, for example, excluding a presentation layer. Further, the web browser may be programmatically configured to not load or parse information that is not critical to the search function. For example, the browser component may be configured to not load media data found on web pages.

In step 550, new media links on each media site that match the MIME format specified in step 510 are added to a database. New media links refers to media links that do not already existing in the database from, for example, a previous execution of the flow process. In step 555, metadata is extracted from each new media link found in step 550. The metadata may also be stored in the database, with a reference to the URL the media that the metadata refers to. An exemplary database is provided with media and metadata table 247 (FIG. 2). An example of metadata is the URL of the Web Page that provided a link for each media URL.

In step 560, the URLs of all new internal media site links on the media site currently being parsed are added to the URLS data structure. New internal media site links

refers to URLs of media sites that do not already exist in the URLs data structure and that are not in the parsed URLs data structure.

In step 570, the currently parsed page is removed from the URLs data structure and added to the parsed URLs data structure. The flow process returns to step 530.

5 C. Verification and Extraction Flow Processes

FIG. 6 illustrates a flow process that verifies and extracts metadata from Internet streaming media files. While media links are specified, other embodiments of the invention may employ the flow process of FIG. 6 within a system that incorporates verification and extraction of any content or resource associated with links stored in a database. A specific application employs a process such as described with FIG. 4 in the system 200. In the system 200, flow process of FIG. 6 may be performed by the AMVME module 240.

In step 610, a module operating the flow process of FIG. 6 fetches or receives an unverified URL from a database. An example of such a database is provided by media and metadata table 247 (FIG. 2). The unverified URL corresponds to a media link on a media site stored in a database such as the media site database 243 (FIG. 2).

In step 620, a determination is made as to whether a URL for a media link is present. If the URL to the media link is not present, the module assumes all media links have been determined as being verified, and the process is done.

If a URL exists, the module in step 630 loads the URL into an Internet multimedia playback software component and programmatically control the component to provide metadata embedded in the media file. In response to this request, the component loads some or the entire file over the Internet and provides the process with this information. In step 640, a determination is made as to whether the media or web resource associated with

60177386-012400  
0044240-98227409

the URL was successfully loaded over the Internet by the module. If the determination is that media was not loaded, then in step 650 the URL associated with the media link is marked as unavailable and verified. The media is marked as verified to prevent the process from revisiting it once it already extracted metadata for it and verified it. The availability mark assists the flow processes described in FIG. 7 and FIG. 8. The determination may be in the negative if, for example, the media link is old and no longer contains a media file that is available for playback through its URL, or if the media link contains content other than what is designated as media.

In an embodiment, the process may use an availability rating for each media.

Under such schema, each media is assumed to have the maximum availability score. For media that are currently not available for playback, step 650 may lower the score by one. The system may consider the media as unavailable if its score is below a predefined threshold. This process is useful since Internet streaming media availability may vary according to factors such as Web server load and the time of day or year.

If the media is loaded, then in step 660 the media metadata is extracted from the playback component. Examples of metadata that can be extracted in this step include artist name, playback duration, playback quality, frame size etc.

In step 670, extracted metadata is stored in a database with the associated URL of the media link that was presently verified. In step 680, the URL of the media link presently verified is marked as verified in the database. The flow process then returns to step 610.

FIG. 7 illustrates a process for interactively adding metadata to URLs of media stored in a database. In an embodiment, the database may correspond to verified URLs of media links determined in FIG. 6. For example, a process such as shown by FIG. 7 may be performed on information stored by AMVME module 240 in media and metadata table

243. The process of FIG. 7 may be performed by a module interface, such as editor interface 275 (FIG. 2).

In step 710, an unedited media record is fetched or received by an interface module from a database. The unedited record may be one or more categories of metadata and other information about a media link, media site, or web resource. In an embodiment, the unedited media information includes a URL to a media link, as well as metadata extracted programmatically, such as described with FIG. 6. In step 720, a determination is made as to whether a record was received. If no record is received in step 720, the flow process assumes there are no unedited records remaining in the database, and the process is done. If a record is received, the in step 730 the interface module is updated with the record received. This may correspond to displaying the record fields to the editor operating the editor interface.

In step 740, the web resource associated with the record received is loaded into an Internet multimedia playback software component. Preferably, the software component is programmatically constructed and controlled by the module interface. The software component plays back the media to the editor. The editor is able to experience the media played back from the web resource associated with the media link. The editor is able to determine metadata information regarding the web resource. For example, the editor may determine mood, genre, quality, appropriate mix name, and description of a web resource such as an audio media creation or a home movie clip.

In step 745, a determination is made as to whether the record received also includes previously determined metadata. The previously determined metadata may be extracted programmatically in another process, such as described with FIG. 6. If the determination is made that the record received does not contain extracted metadata information, then in step 750 the editor interface automatically extract metadata from the



web resource associated with the URL of the record. To accomplish this step, the editor interface may access another module that automatically extracts certain types of metadata information. For example, the editor interface may forward the record to AMVME module 240 (FIG. 3) that performs a flow process such as shown by FIG. 6.

5        Once metadata is included with the record, then a determination is made in step 760 that the editor operating the interface module may choose to save the auto-extracted metadata already included with the record in the database. If the determination to step 760 is negative, then in step 765 the editor updates media information with editor provided metadata using input elements on the editorial software user interface. Once the  
10        determination in step 760 is positive, the record is marked as edited and updated in step 770. Then, all the newly added metadata for the media record is updated to the database. The process then continues from step 710 for the next unedited media. The media is marked as edited so it won't be included for editing in the process.

FIG. 8 illustrates a process for generating a play-list. The flow process of FIG. 8  
15        assumes that play-lists names are predetermined and stored in a database. Each play-list is identifiable by its name. For example, classic music, jazz and rock. The play-lists include records of media links. In an embodiment, a record includes at least a URL to a streaming media that is categorized in a database as belonging to the play-list name. In step 810, a play-list name is received from the-play-lists database. In step 820, a determination is  
20        made as to whether a play-list name was received. If the determination is negative, the system assumes it produced play-lists for all play-lists names, and the flow process is done. This process is routinely executed to add all newly added media to the appropriate play-lists.

In step 830, media records that match search criteria are fetched from the media  
25        database. The criteria is that each record play-list record field must match the current play

list name obtained in step 810 and that the 'in-play-list' record field for the play list name has False value. In step 840, a play-list is generated to include all media stored at the records fetched is step 830. The new play-list contains one record entry for each fetched media record. -Preferably, each play list record includes media URL and metadata information that is obtained from the media database record.

In step 850, the media records called in step 840 are labeled as being in the "in-play-list" for the current play-list name. This is achieved by setting the "in-play-list" value for the media database record to true for the appropriate play-list name.

In step 860, the generated play-list is made available to a server-side module, such as web server module 270 (FIG. 2). As an example, the play-lists may be stored, copied or appended to be made available on the web server module. Once the play-lists are available on a server-side module, media URLs and metadata stored in the play-lists is made available to the user terminal so that the user terminal may customize media output available from the play-list. Specifically, one or more playback applications that run on the user terminal may read the play-lists, access media links on the play-lists, continually play-back streaming media from media URLs in the play-list and present media metadata to users for further interaction. The play-list may also be configured to provide access to a client-side media player component that uses a URL of a media link to load and play the media associated through it. Additionally, users may further modify and edit play-lists to create personalized media programming.

FIG. 9 is a flow process for software or hardware application that enables a user terminal to playback streaming media programming determined from pre-generated play-lists. The play-lists are predetermined by, for example, a process described with FIG. 8. Alternatively, play-lists may be manually generated. An embodiment described below assumes that play-lists are available for the process, and that play-lists are identifiable by

play-list names. The playback component can access the play-list without any direct interaction with server side modules.

In an embodiment, the flow process employs a streaming media player component installed on the user terminal. The media player may be preexisting on the user terminal.

5 Examples of media players for use with an embodiment include RealNetwork RealPlayer™, Microsoft Windows Media Player™, and Apple QuickTime™. The application described in the process may be web based or installed on the user terminal.

In step 910, an application interface for a media player on the user terminal is provided.

10 In step 920, a default play-list name is selected from a list of play-lists. In an embodiment, a database of play-lists is stored on or accessible through the web server module 270. The play-list generator module creates and stores play-list on the web server module to provide the interface with the user terminal 210 and media player stored thereon. Each play-list may store two or more media links, and preferably a plurality of  
15 media links.

In step 930, one or more play-lists for the current play-list name are loaded. In step 940, media is presented and played-back on the user terminal. To playback media, server-side modules may provide the media player with URLs to media links that are stored in each play-list. Each play-list may include one or more media link URLs and media  
20 metadata. The user terminal then accesses the URL and loads the web resource associated with that media link into a streaming media playback component on the user terminal. Media playback on the user terminal includes outputting, for example, audio and/or video stored in digital format on web resources associated with a media link. In embodiments where the web resources include video media, the step described may dynamically adjust

the interface size and the playback component that will handle the actual playback according to the media clip metadata.

Under an embodiment of the invention, the media playback component continuously plays back media by (i) accessing a first site on the network and playing back  
5 media from the first site, (ii) then automatically accessing a second site and playing back media from the second site. The media sites may be provided by play-lists which that are made accessible to the media playback component. The sequence for automatically and continuously playing back media may be repeated for each media link included in the play-list. The play-lists may include hundreds of media links, thus allowing the media  
10 play back component to automatically and continuously play back media using numerous sites on a network. As a result, a user is able to experience continuous media play back for hours at a time.

The user terminal also includes an interactive interface to affect the media being played back. An end user on the user terminal may choose to manipulate media playback  
15 through one or more commands that may be inputted through the application interactive interface or presentation layer.

In step 945, a determination is made as to whether an event occurred. If an event occurred, the flow process determines the event. The flow process may determine the event that occurred sequentially or concurrently. Preferably, the flow process is  
20 configured to receive media playback event from the streaming media playback component. If the event is to skip the currently played media, then a determination in step 960 causes a corresponding action of the media being skipped, and the process step returning to step 940. If the process received a playback error event, or an playback error event, then a determination in step 965 causes the flow process to return to step 940 to  
25 playback media from the next Web resource of the play-list or play-list name.

If a user quits the application, or signals to quit, then a determination in step 970 causes the flow process to be done. In step 975, a user may choose to view a new media Web site while media is being played back on the user terminal. Then in step 980, the media Web site is opened in a new Internet window, preferably using the client side Web browser.

In step 982, a determination is made as to whether the action selected by the user on the user terminal is to send an e-mail message to one or more e-mail addresses that allow the receiver(s) to playback the current media and the current play-list played by the sender at the time of the event. A user may send either an e-mail containing the URL, the play-list, the play-list name or a URL link. When the message receiver clicks on the link, his terminal will execute the media playback application and will start playing back the media and the play-lists played at the time of the message-sending event. If the determination is positive, then in step 984 the user terminal prompts the user for an one or more e-mail addresses, and then prompts the user to transmit the email. The user need not have an e-mail client software application for the operation to succeed. The e-mail may be directed to terminals having a streaming media playback component and Internet access. Preferably, the e-mail message is directed to a user having a user terminal that communicates with server-side modules so that the recipient user terminal is automatically plays back media from the transmitted media link received upon the email being opened.

In step 986, a determination is made as to whether a user wishes to rate a media played back from the media links. If the determination is positive, then the user is prompted to rate the media in step 988. A rating value is transmitted to a backend web rating system application using the Internet. This operation is part of the service rating system that includes server side modules that produce, in combination with this operation, top and bottom rating media in each media category.

Step 990 illustrates other exemplary actions that may be received from a user on the user terminal interfacing with the playback application. For example, user actions may correspond to pausing media playback, adjusting volume, picture controls, size, seeking within the media, etc. In step 992, playback settings are changed according to user input.

5 In an embodiment, the flow process returns to step 945 to check for another event if any of the determinations in step 982-992 are negative.

#### D. Rating System for Media Play Back

An embodiment of the invention includes a rating system with a media search and playback system. The rating system allows users to listen or view media segments

10 available over the Internet according to a rating. The rating may be determined by input received from users. The rating of media may be categorized into genres or categories.

FIG. 10 illustrates an architecture for use with the rating system, under an embodiment of the invention. The rating system 1000 may be employed with, for example, the system 200 (FIG. 2). The rating system 1000 may include or cooperate with

15 components of a system such as described with FIG. 2 to enable the user view and/or select media clips from play-lists. The generated play-lists may contain a list of links to media on the Internet. Selection of media clips by the user causes media to be played back to the user over the user terminal.

The rating system 1000 includes a backend database management component

20 1045. The database management component 1045 maintains organizational data structures such as tables that describe rating information for media clips. The media clips include Internet streaming audio or video. The rating information may be in the form of values such as, for example, total votes counted. In an embodiment, database management component 1045 maintains records that comprise meta information on each media clip

including the URL to the media clip, the current rating of the media clip, and the total votes for that media clip.

A user 1010 on a user terminal interacts with a web-based playback interface 1020. As an example, play-back interface 1020 outputs play-lists 1018, 1022 to the user. The media clips in each play-lists may be outputted automatically, or displayed for the selection of the user. The play-back interface 1020 may also display to the user genre field 1016 or category field 1014 of the selected media clip, or play-list 1018, 1022. The playback interface 1020 includes features to enable a user on the user terminal to make entries or selections regarding preferences and opinions, as well as other types of information. The user may also view ratings stored on backend database 1045. The user may enter selections by, for example, using icons or other display features. The user may make entries by, for example, inputting text or voice. FIG. 10 illustrates a rating selection component 1012 as a feature of play-back interface 1020. As an example, the rating selection component 1012 allows users to rate a media output between a scale of 1 to 5.

A user 1010 may input a rating to play-back interface 1020. The rating is signaled from play-back module 1020 to a rating module 1030. In an embodiment, rating module 1030 maintains a tally for each media clip. The tally compiles ratings received from play-back module 1020. The ratings may be received from more than one user and/or user terminal. The tally may be implemented through a protocol that enables the rating module 1030 to organize media clips according to an order. The organization of the media clips may correspond to a user preferential list, where preferred media clips are, for example, listed together or listed before less preferred clips. The rating module 1030 may also determine a genre, category, or other organization information through selections or entries received from the play-back module 1020. The selections may be tallied through any protocol, such as summation, averages, weighted averages and moving averages. In

another embodiment, the rating module 1030 may maintain a text field to store user comments regarding each media clip.

In an embodiment, the rating component 1030 updates the rating information maintained in the database management component 1045. For example, the rating  
5 component 1030 may update values of the current rating and total votes for each media link.

A play-list generator 1040 generates play-lists based on rating information maintained in the database management component 1045. The play-list generator 1040 may signal to retrieve or receive records for each media clip. The play-list generator 1040  
10 then automatically generates one or more play-list 1042. As previously discussed, each play-list is a list of media links. In an embodiment, the play-lists 1042 are generated according to the current rating and/or rating for each media clip. The generated play-lists are provided by the play-list generator 1040 for the play-back interface 1030.

The user 1010 may choose to listen to play-lists containing media clips rated  
15 according to one or more criteria. The play-lists may also be organized according to other factors, such as genre and category.

FIG. 11 is a flow chart that allows a user to listen to media clips that are rated according to one or more criteria. In step 1110, the user is provided a user-interface that allows users to receive media sorted according to one or more categories. The categories  
20 correspond to genres, such as type of music etc.

In step 1120, the user selects a category from the options presented by the user-interface. In response to the selection, the user terminal is provided one or more play-list in step 1130. The play-list received by the user-terminal matches the category or genre selected by the user. Further, play-lists contain predetermined media links to media clips.  
25 The media clips in each of the play-lists are determined according to a rating system,



using a system such as described by FIG. 10. The predetermined play-list may correspond to a play-list generated by play-list generator 1040 (FIG. 10). Once the play-list is received by the user terminal, the flow process returns to step 1120.

In step 1140, media clips are played back on the user terminal. The media clips are played back consecutively and automatically, so that the user experiences continuous media playback. For example, the play-lists may contain numerous media creations from a selected genre. The media creations may be determined for the play-list according to a rating formula. The user is provided the media creations of the selected genre continuously, so that the user's media experience resembles listening to an album.

FIG. 12 illustrates a flow process for updating a rating of a media clip, under an embodiment of the invention. In step 1210, a module is provided a rating event. The rating event is a rating for a particular media clip, having an associated URL. The rating from the user is predefined from a closed set. For example, the user may provide a rating from 1 to 5.

In step 1220, a record is located for the media clip that was currently rated. The record may be stored in a database, and include the media link for the media clip, the current rating of the media clip, and the votes received for that media clip. In an embodiment, the record may include more than one URL associated with the media clip that was rated. In an embodiment, the record is maintained in database management component 1045 (FIG. 10).

In step 1230, a rating field for the media record is updated. The rating field may correspond to the current rating of a media clip. A module such as the rating module 1030 may update the rating field in database management component 1045 (FIG. 10). The media record is updated to determine a new rating. In one embodiment, the new rating is

an averaged based formula. The formula may also be weighted. An example of a formula to determine a rating, under an embodiment of the invention is:

$$\text{Newrating} = 1/(n+1) (N * (\text{old rating}) + \text{user provided rating})$$

N is the total number of votes received, and newrating ranges between 0 and a maximum value.

In step 1240, record for the media clip rated is further updated to add an additional vote to the field for votes received. The flow process then returns to step 1210.

FIG. 13 illustrates an exemplary structure for a database to maintain updated records on ratings and votes tallied. The table may associate values corresponding number of votes, rating, and other information to a media link containing a media clip.

FIG. 14 illustrates a flow process for generating media clips into play-lists according to a rating criteria. Play-lists including a rating criteria are referred to as rated play-lists. The flow process assumes known categories for media clips. The flow process also assumes a rating for rated media clips, and the number of media clips in a rated play-list. The flow process may be used with any of the aforementioned embodiments.

In step 1410, a next category may be fetched from a database containing the different categories of media clips. In step 1420, the system makes a determination as to whether a category was received. If no category is received, the system assumes that there are no more media categories to be rated.

In step 1430, a new play-list is created for a current category. In step 1440, up to N rated media clips from a database of rated media clips are added to the play-list.

Preferably, N is a constant in the flow process. Then in step 1450, an old play-list is

deleted, and the new play-list is saved. The new play-list may be saved in a format that

follows predefined protocol so that the play-list and its contents are accessible to a streaming media play back interface. The flow process then returns to step 1410.

FIG. 15 illustrates a flow process for programmatically categorizing media files. The process assumes a database containing metadata associated with media clips. The metadata includes metadata provided by a human editor. For example, the metadata may pertain to categories such as genre, mood and atmosphere.

In step 1510, a record is retrieved from the database. The record is retrieved with metadata information containing a first type of metadata information and a second type of metadata information. As an example, the first type of metadata information may correspond to a genre of music, and the second type of metadata information may correspond to an artist. In step 1520, a determination is made as to whether a record was received. If the determination is negative, then the process assumes that all media clips have been categorized.

If the determination in step 1520 is positive, then all records in the database having the first type of metadata information are retrieved in step 1530. In step 1540, all records retrieved in step 1530 are updated to include the second type of metadata information. As an example, all records belonging to a particular artist (first type of metadata information) or given additional metadata information of a particular genre (second type of metadata information). The process then returns to step 1510 to retrieve another record.

In one embodiment, the second type of metadata information is a genus category, and the first type of metadata information is a species of the first type of metadata information. Once the first record is known to have the a particular species and genus, the genus may be determined and stored for all records having the same species.

#### E. Personalized Media Playback

FIG. 16 is a flow process to create personalized play-lists of streaming media files available on the Internet (or other networks). The play-lists may be personalized by users on user-terminals.

5 In step 1610, a user chooses to add a URL of a selected media clip to a personal favorite play-list. In step 1620, the flow process adds the URL (and metadata) of the selected media clip to a user terminal store for user persistent information, such as an Internet cookie. The persistent data store is then accessible for the web-based play back application on the user terminal.

10 In step 1630, the user selects to play back media clips from that user's favorite play-list. In step 1640, the system reads back the media clips from the persisted data store. In step 1650, the system plays media clips using a URL associated with each media clip. The cookie may also provide additional URLs. Thus, multiple media clips may be played continuously from different sites on the Internet.

#### 15 F. Distributed Architecture

An implementation under an embodiment provides a distributed architecture in which media may be accessed from a plurality of sources by one user terminal. The distributed architecture inverts conventional media distribution paradigms. Numerous streaming media files can be streamed to an individual user terminal continuously from  
20 throughout the Internet using the embodiment of the distributed playback architecture. The distribution architecture is scalable to provide thousands or millions of streaming media files to user terminals. The users can then play media files located throughout the Internet in a continuous manner from the numerous Internet sites.

Among other advantages, the distributed architecture permits simultaneous playback of, for example, thousands or millions of multiple streams which do not congregate on a single point. This avoids congestion arising under examples of the current media paradigms. This ensures that the embodiment of our distributed architecture may  
5 "scale," or permit the simultaneous playback of, for example, thousands or millions of simultaneous streams. Further, the quality of the user experience is not affected by scaling a system under the distributed architecture embodiment.

In contrast, conventional broadcasting employs one radio or television signal to broadcast to listeners or viewers. Media files disseminated over the Internet today may be  
10 distributed in a manner which is somewhat similar in that the media file is located on a single server (or small group of servers) which is accessed by potentially large number of Internet users. As a result, the experience of the users may diminish due to the limited ability of current systems to scale.

This distributed playback architecture, the delivery of streaming media through this  
15 playback architecture, in combination with the search functionality performed by the back-end module, and the rating and personalization features of the playback client terminal module permits the creation of a broadcasting system that is personalized by an end user. A personal broadcasting system permits each individual user to create media programs which can be sent to, for example, thousands or millions or other users who can  
20 simultaneously play different programming combinations using a distribution of Internet (or other network) sites.

5 invention to the precise forms disclosed. Many modifications and equivalent arrangements will be apparent.



6017786-012400

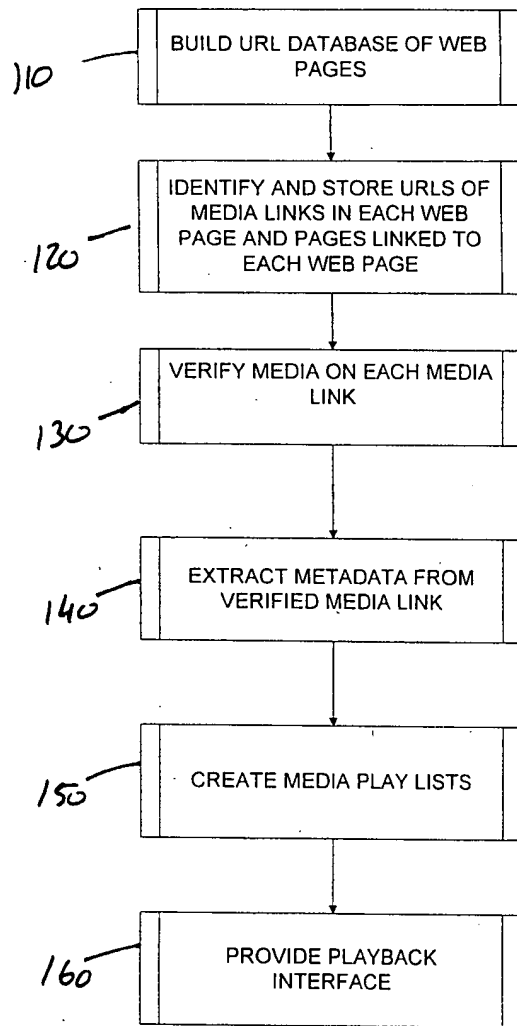
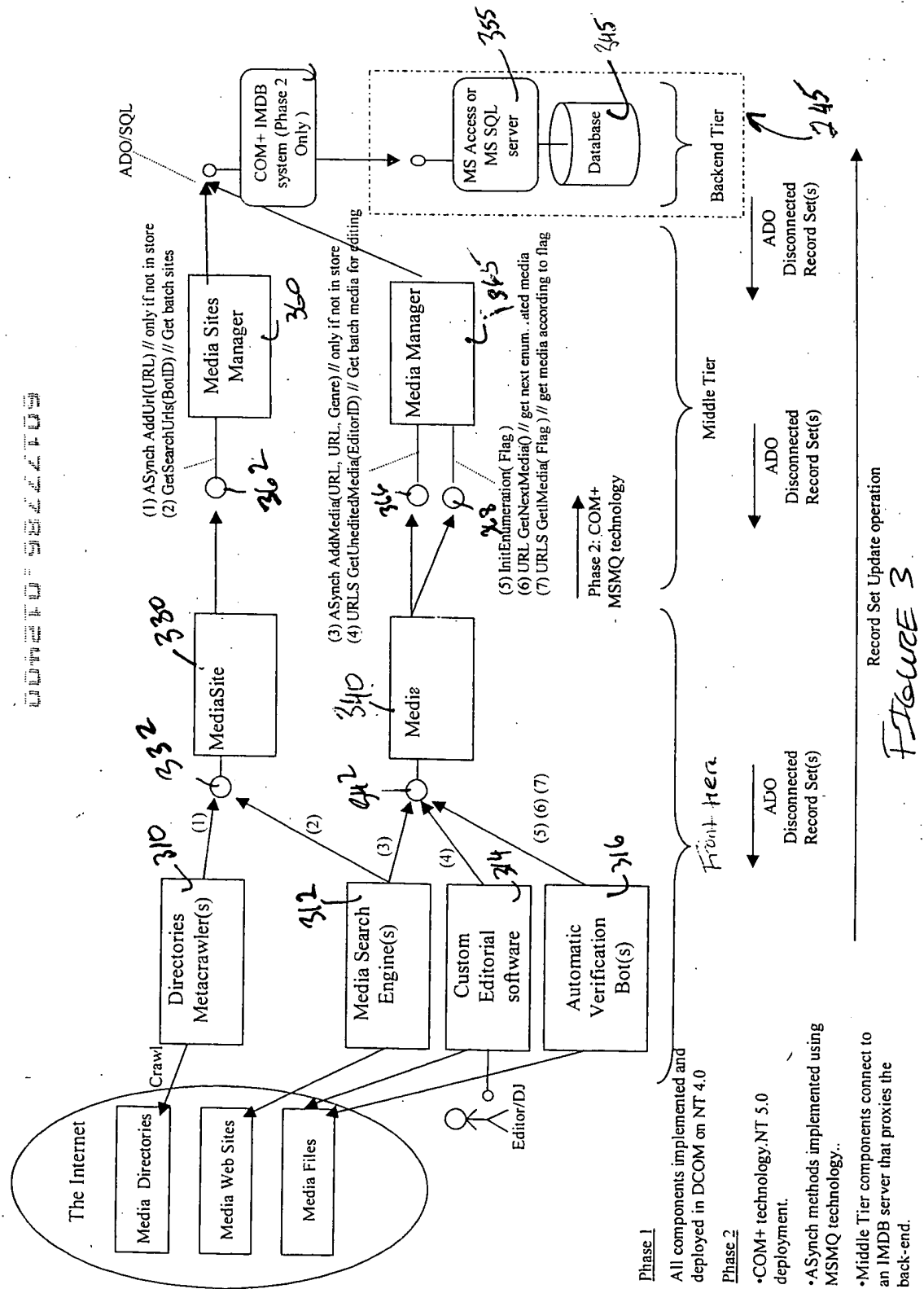
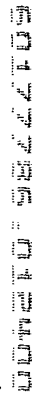


FIGURE 1

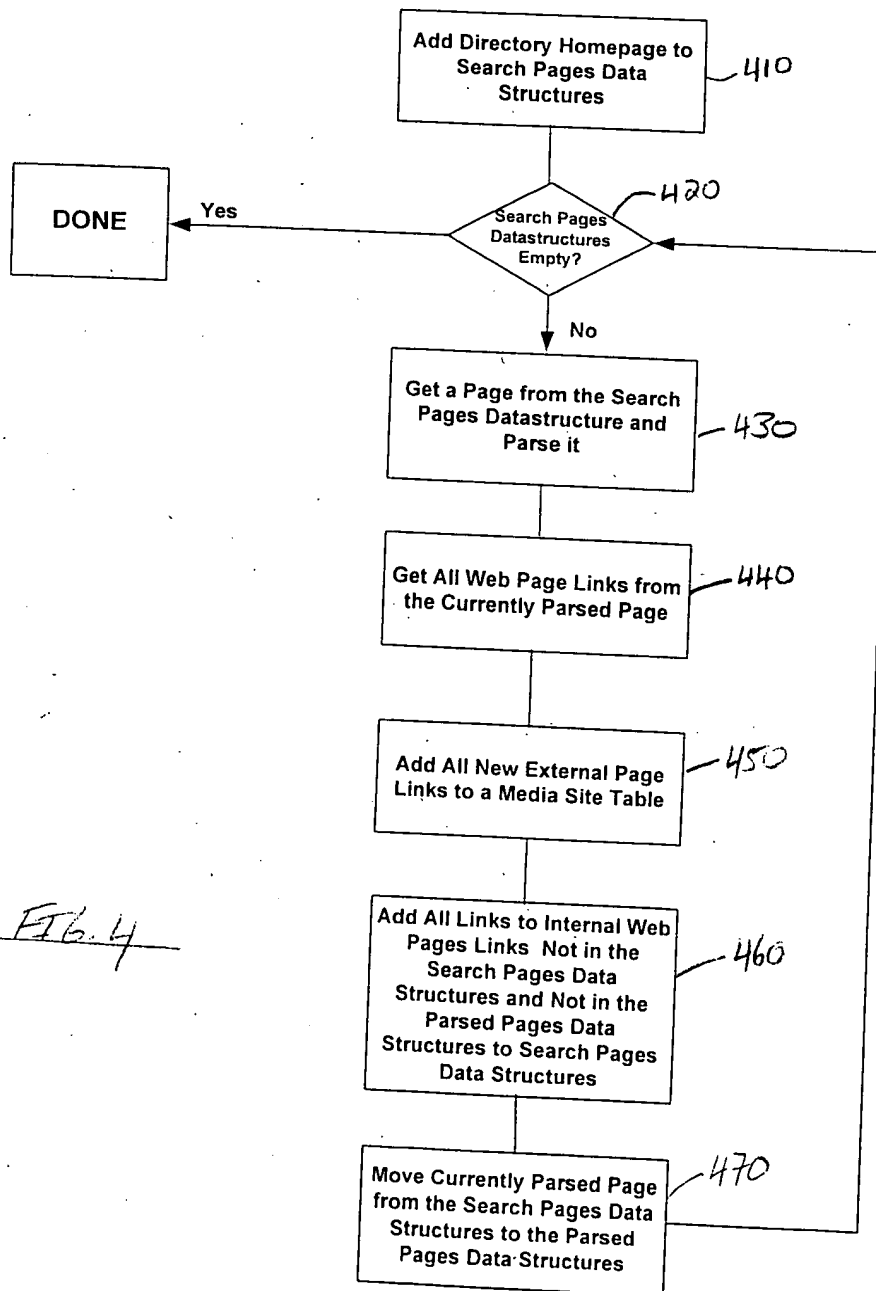








6017786-01400



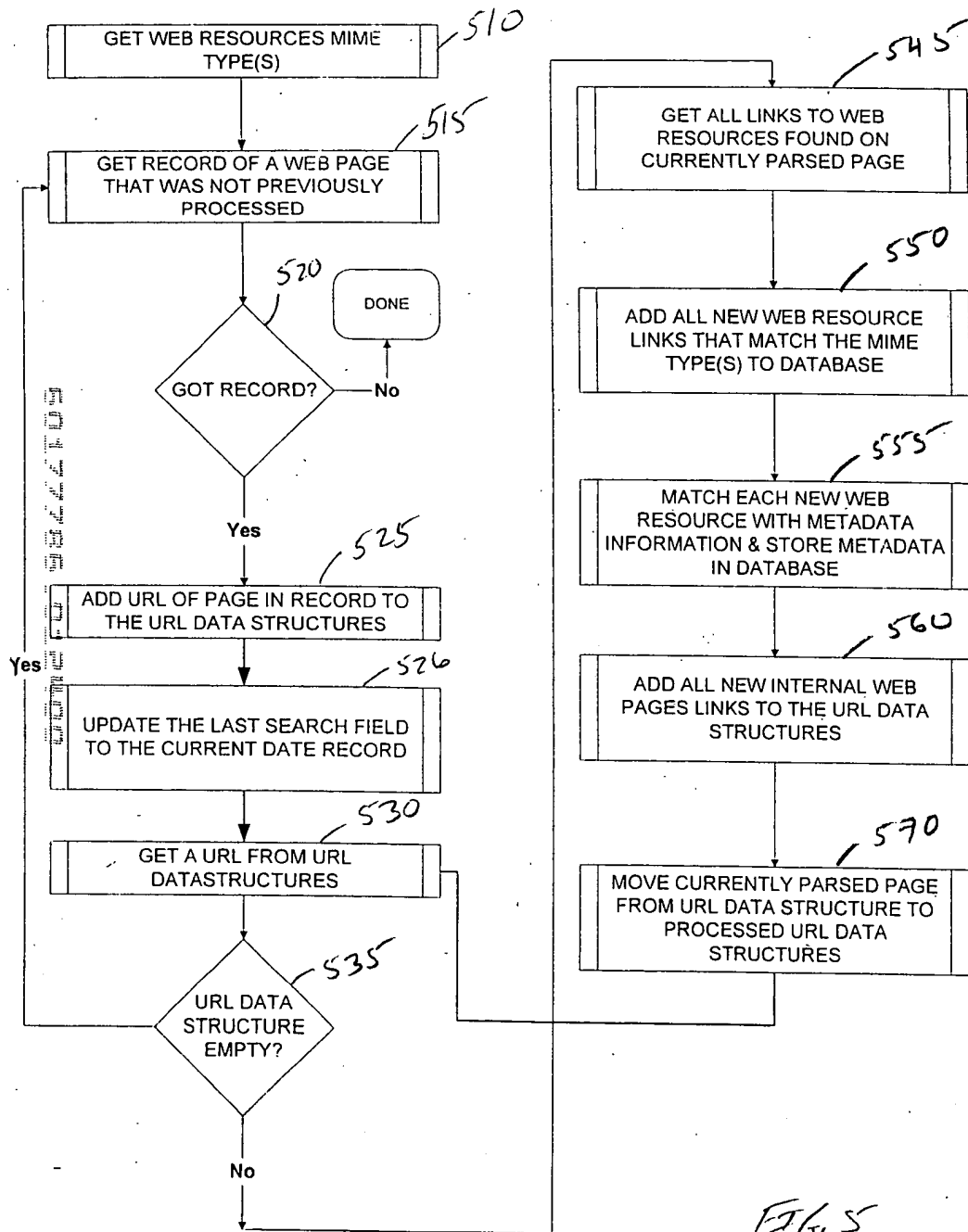


FIG. 5



60477785-012400

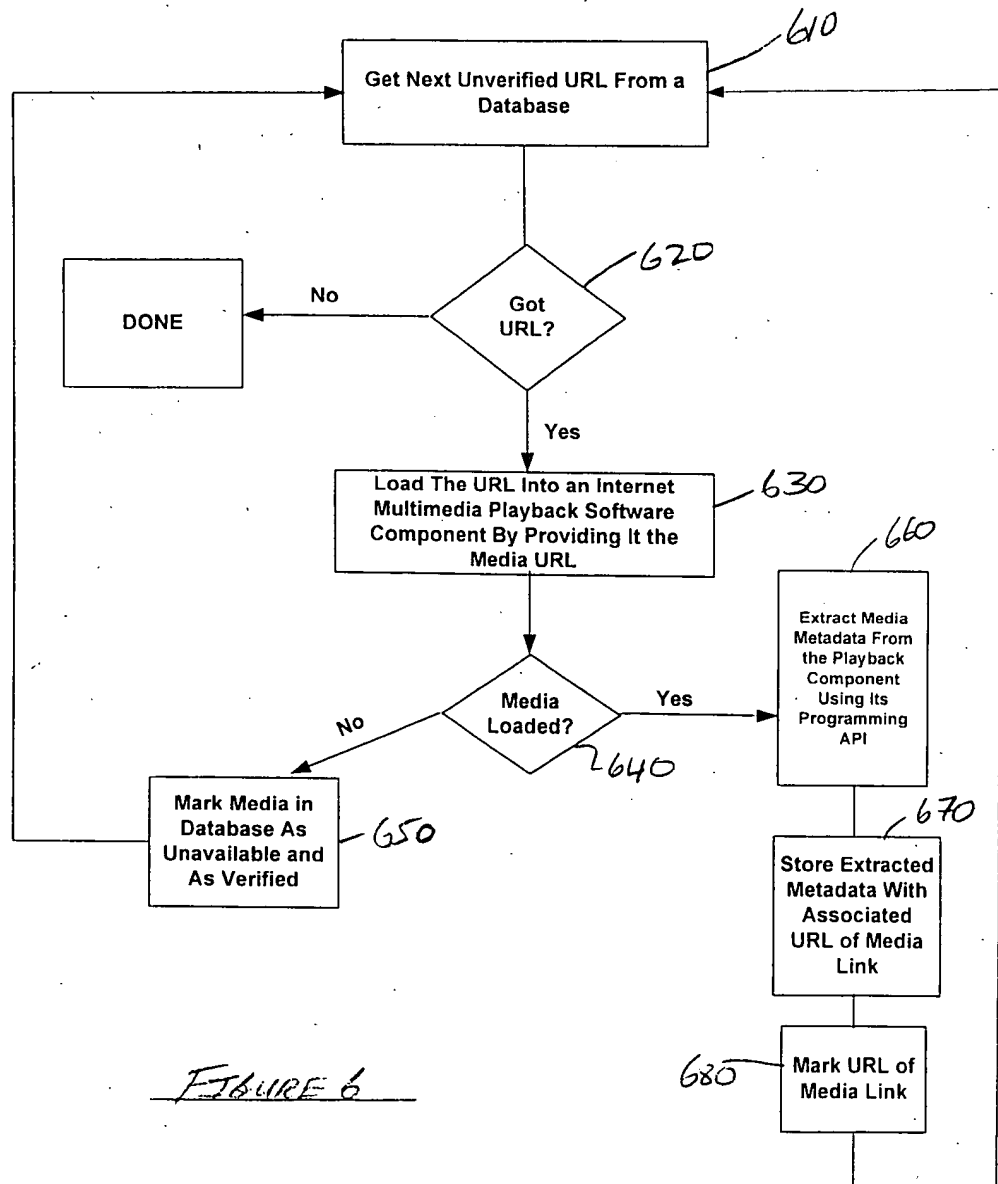
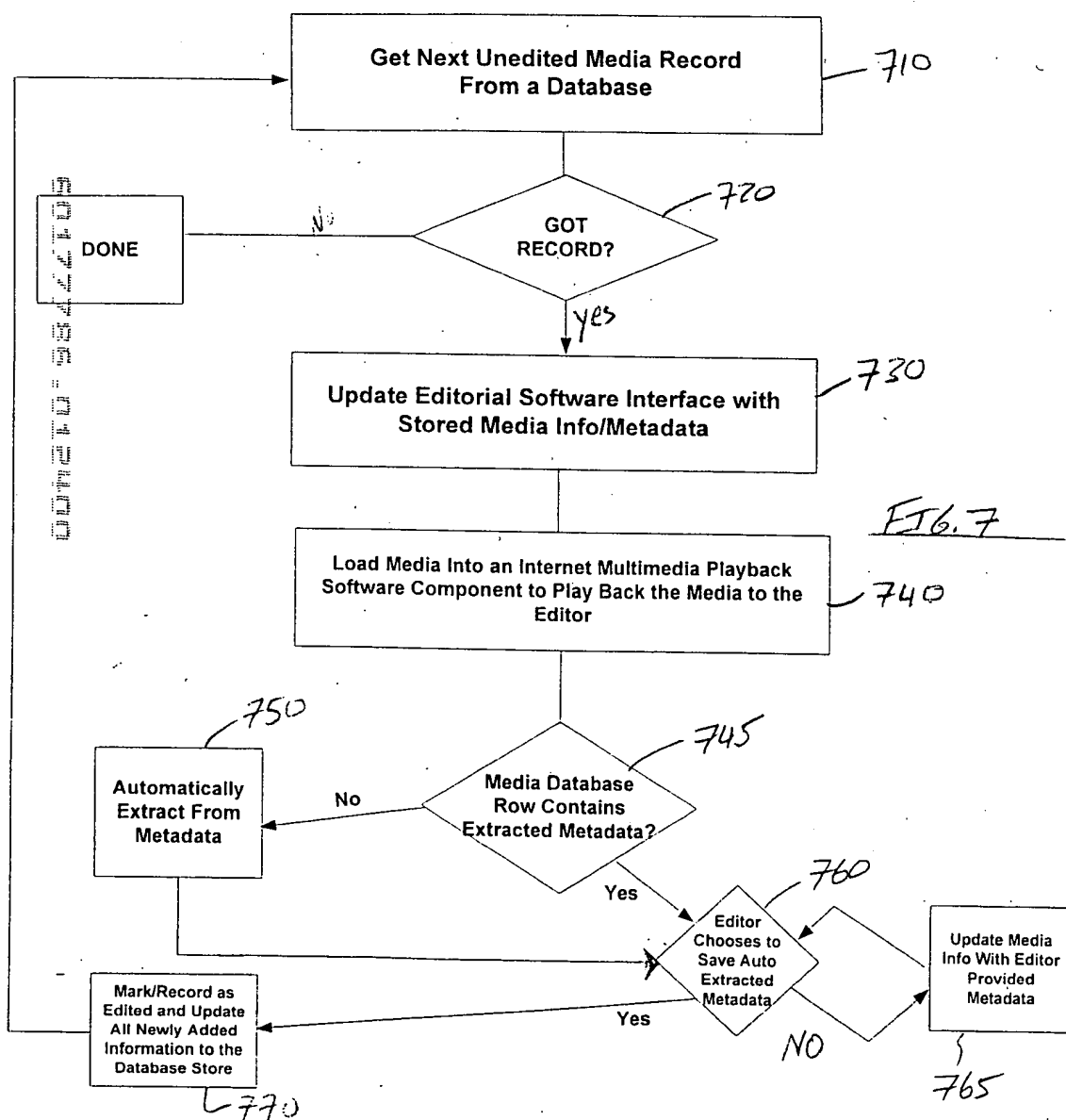


FIGURE 6



FT6. 7

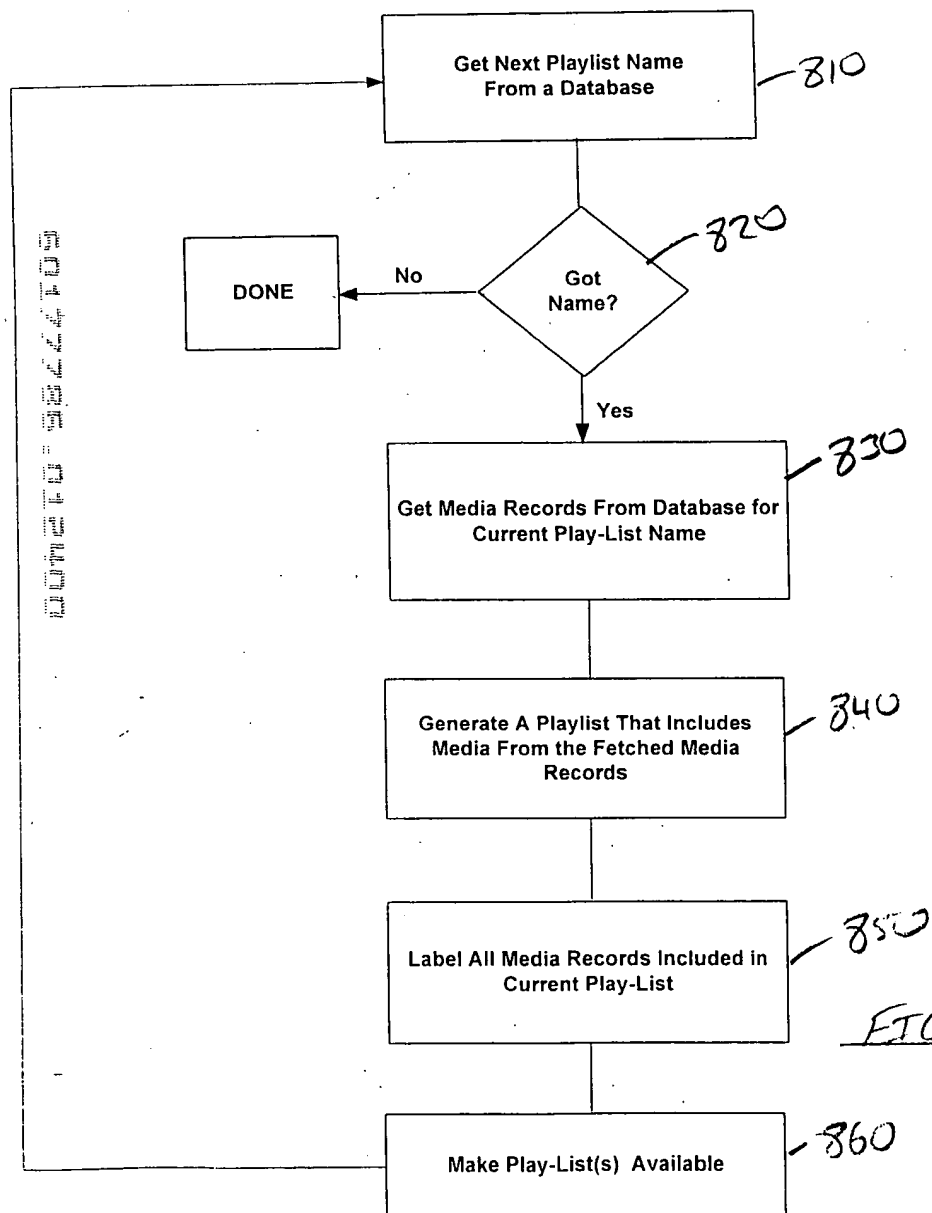
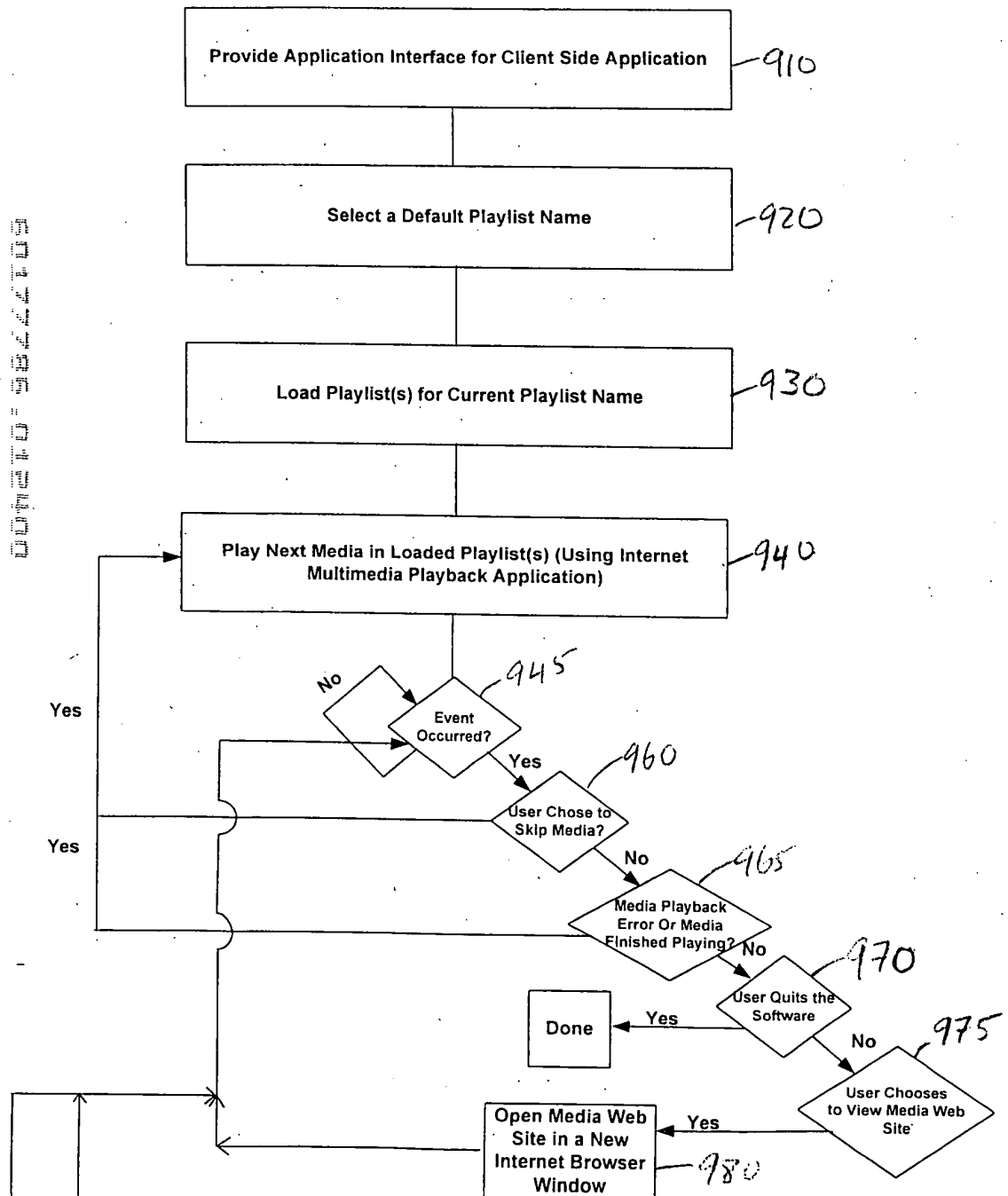




FIGURE 9



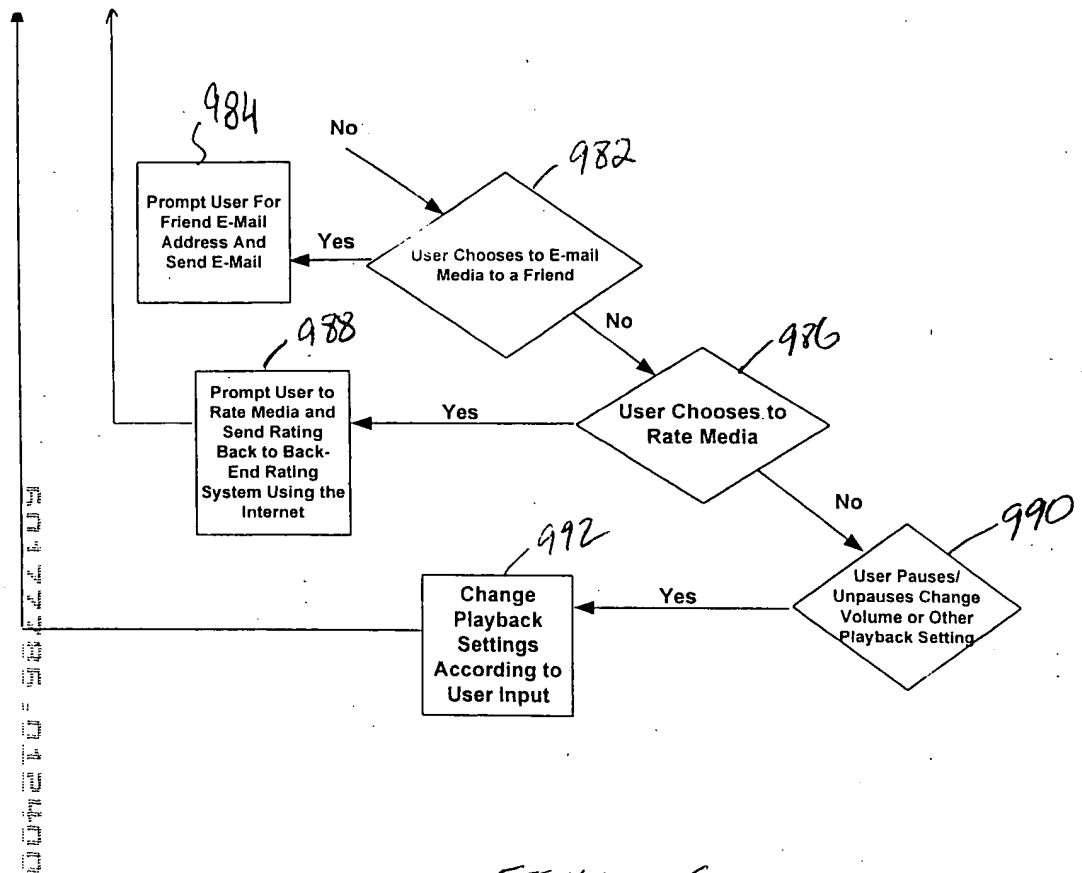


FIGURE 9 (contd.)



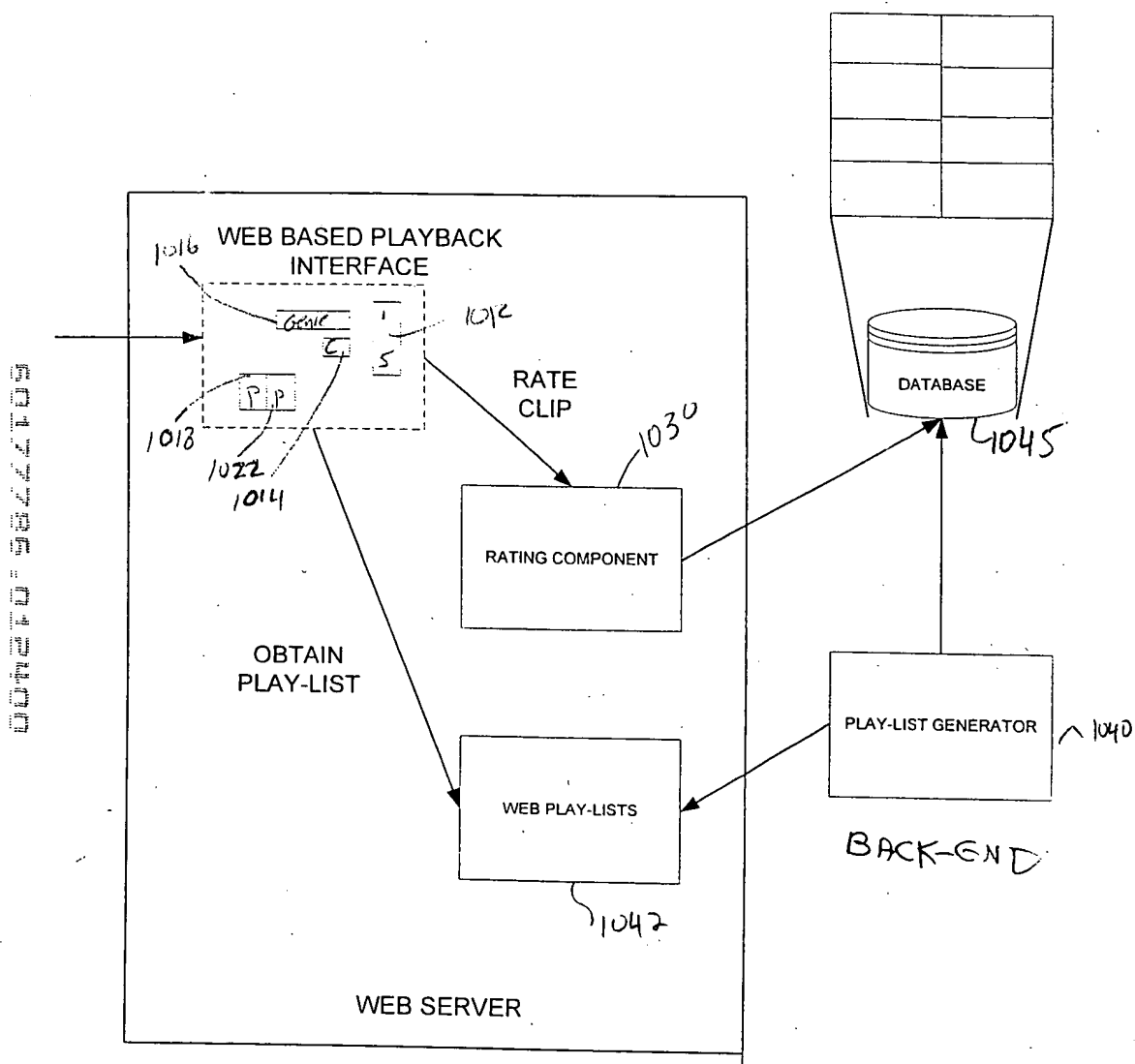


FIGURE 10

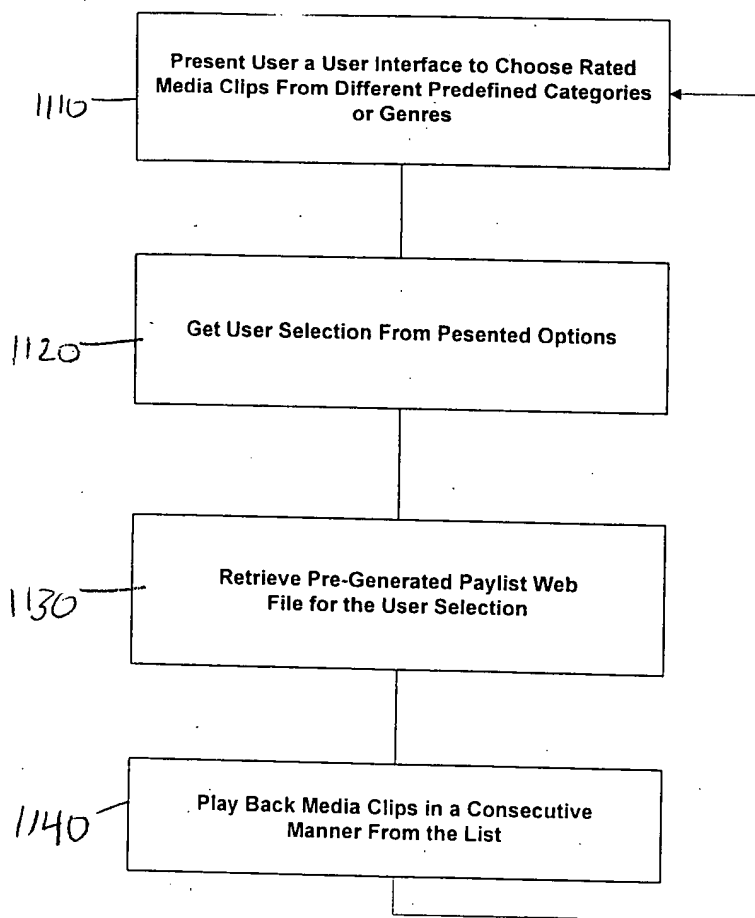


FIG 11



6017786-012400

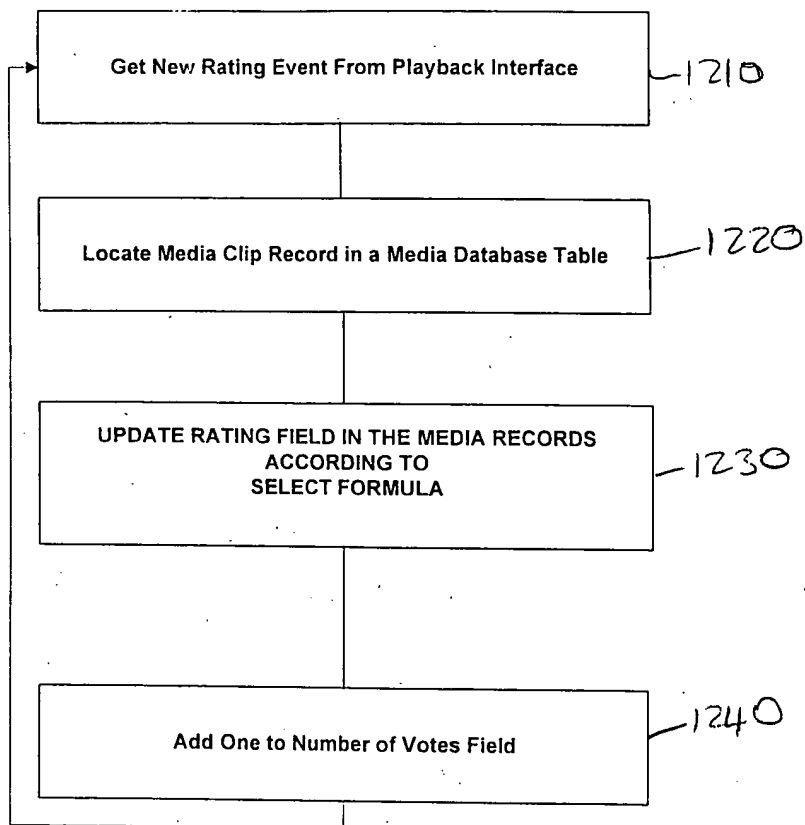


FIG. 12

MEDIA DATABASE TABLE

FIG. 13

Media URL	Number of Votes	Rating	Additional Info
URL	0 ... N	0 ... Max Rating	



004210 5822209

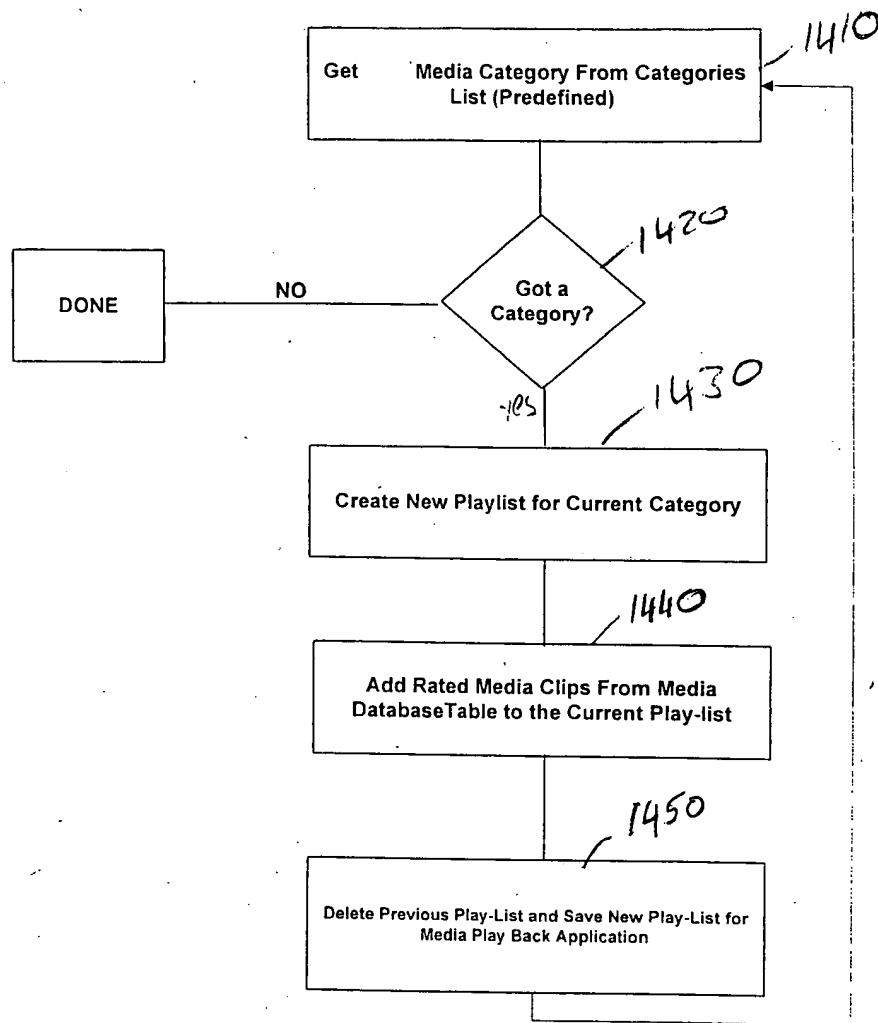


FIG. 14



50177285-013400

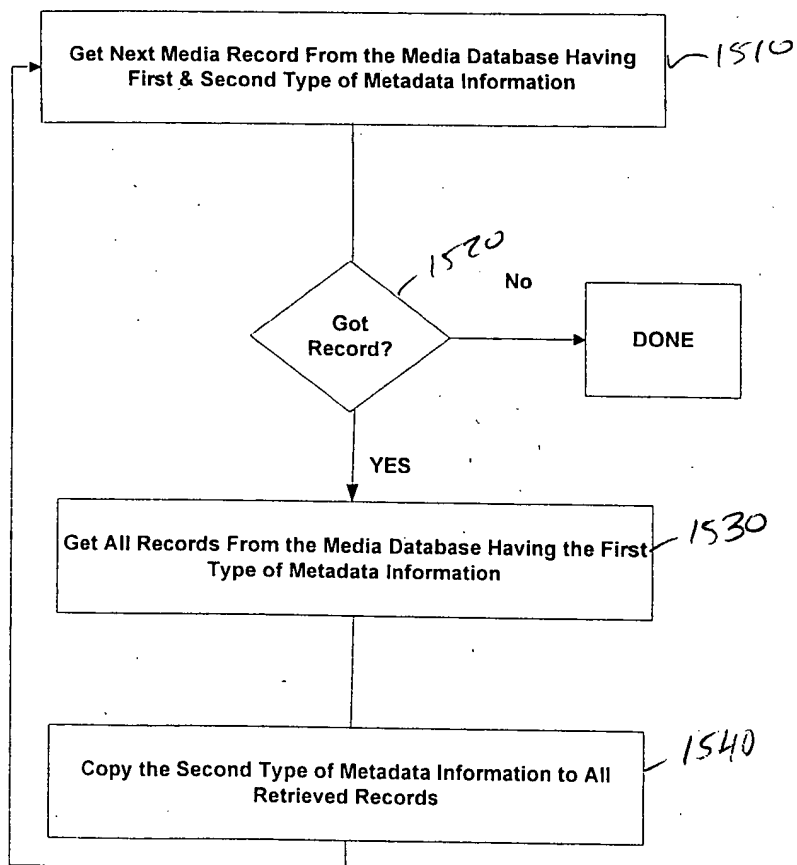


FIG. 15